Graph-Embedding Empowered Entity Retrieval

Emma J. Gerritse, Faegheh Hasibi, and Arjen P. de Vries

Institute for Computing and Information Sciences
Radboud University, Nijmegen, the Netherlands
emma.gerritse@ru.nl f.hasibi@cs.ru.nl a.devries@cs.ru.nl

Abstract. In this research, we improve upon the current state of the art in entity retrieval by re-ranking the result list using graph embeddings. The paper shows that graph embeddings are useful for entity-oriented search tasks. We demonstrate empirically that encoding information from the knowledge graph into (graph) embeddings contributes to a higher increase in effectiveness of entity retrieval results than using plain word embeddings. We analyze the impact of the accuracy of the entity linker on the overall retrieval effectiveness. Our analysis further deploys the cluster hypothesis to explain the observed advantages of graph embeddings over the more widely used word embeddings, for user tasks involving ranking entities.

Keywords: Entity Retrieval · Graph Embeddings · Word Embeddings

1 Introduction

Many information needs are entity-oriented, and with the rise of knowledge graphs in Web and enterprise search [20], the role of entities has gained importance, both in the UI/UX where so-called entity cards are shown in response to entity-oriented queries, and in the ranking, where presence and absence of entity mentions is weighted differently from traditional term occurrences.

Recently, word embeddings have been shown to be helpful for a number of information retrieval problems. In the case of entity retrieval, a natural representation would however not just represent words in context of their textual neighborhood, but in context of the knowledge graph instead. Here, we would want to apply graph embeddings instead of word embeddings, where the semantic space constructed by graph embeddings does not only encode the textual context of an entity mention, but also the context as defined through the knowledge graph. Considering Wikipedia as the knowledge graph to define the entities of interest, for example, creating a graph embedding representation does not just take the entity's page itself as context, but also its anchor text, presence in lists and/or tables, *etc.* It is therefore likely that graph embeddings capture more of the entity's semantic roles and as a result may distinguish better between ambiguous entities than a plain word embedding based representation.

Exploring the use of graph embeddings in entity retrieval, we have studied a two-stage entity retrieval approach where the second stage employs graph

embeddings for re-ranking the retrieval results of state-of-the-art entity ranking methods. We investigate the following research questions:

RQ1: Does adding graph embeddings improve entity retrieval methods?
RQ2: Which queries are helped the most?

To our knowledge, we are the first to investigate how the structural information captured in graph embeddings can contribute to improved retrieval effectiveness in entity-oriented search. The contributions of this paper are as follows: We have build graph embeddings from Wikipedia as a knowledge graph[1] and evaluated the contribution of these embeddings as a representation of entities in the ranking algorithm, using the DBpedia-Entity V2 collection [12]. For every query, we re-rank the results of state-of-the-art entity retrieval methods using the similarity between the entity embeddings of the candidate entities retrieved in stage one with the entity embeddings of the entities identified in the query (using an off-the-shelf entity linker). We show that re-ranking using graph embeddings improves retrieval effectiveness, and investigate how to explain this result by comparing the structure of the two types of embeddings. We also analyze why some queries are helped by this method while others are not.

2 Related Work

2.1 Word and Graph Embeddings

Distributional representations of language have been object of study for many years in natural language processing (NLP), because of their promise to represent words not in isolation, but 'semantically', with their immediate context. Algorithms like Word2Vec [19] and Glove [21] construct a vector space of word domains where similar words are mapped together (based on their linguistic context). Word2Vec uses neural networks to predict words based on the context (continuous bag of words) or context based on a word (skip gram). These word embedding representations have turned out to be highly effective in a wide variety of NLP tasks.

Word embeddings have been shown to help effectiveness in document retrieval [6,7]. In [7], locally trained word embeddings are used for query expansion. Here queries are expanded with terms highly similar to the query, and it is shown that this method beats several other neural methods. In [6], embeddings are used for weak supervision of documents. This paper uses query embeddings and document embeddings to predict relevance between queries and documents, when given BM25 scores as labels. It is able to improve on BM25.

Word embeddings consider the immediate linguistic context of the word occurrences. Going beyond just the text itself, researchers have proposed to develop so-called *graph embeddings* to encode not just words in text, but words in context of semi-structured documents represented as graphs - for example, to distinguish

[1] Downloadable at `https://github.com/informagi/GEEER`

the occurrence of a word in the title of a document from its occurrences in a paragraph, or in a document's anchor text.

Different methods to produce graph embeddings have been proposed. Methods like Deepwalk [22] expect non-labeled edges and can be considered extensions of the word embedding approaches discussed before. Other approaches include the well-known method Trans-E [4], where edges in the graph are denoted as triples *(head, label, tail)*, where *label* is the value of the edge. Adding graph embedding vectors of the *head* and the *label* should result in the vector of the *tail*. The embeddings here are learned by gradient descent.

Wikipedia2Vec [26] applies graph embeddings to Wikipedia, creating embeddings that jointly capture link structure and text. The Wikipedia knowledge graph is indeed a natural resource for using graph embeddings, because it represents entities in a graph of interlinked Wikipedia pages and their text. The method proposed in [26] embeds words and entities in the same vector space by using word context and graph context. The word-word context is modeled using the Word2Vec approach, entity-entity context considers neighboring entities in the link graph, and word-entity context takes the words in the context of the anchor that links to an entity. The authors of Wikipedia2Vec demonstrate performance improvements on a variety of NLP tasks, although they did not consider entity retrieval in their work.

2.2 Entity retrieval

An entity is an object or concept in the real world that can be distinctly identified [2]. Knowledge graphs like Wikipedia enrich the representation of entities by modeling the relations between them. Methods for document retrieval such as BM25 have been applied successfully to entity retrieval. However, since knowledge bases are semi-structured resources, this structural information may be used as well, for example by viewing entities as fielded documents extracted from the knowledge graph. A well-known example of this approach applies the fielded probabilistic model (BM25F [23]), where term frequencies between different fields in documents are normalized to the length of each field. Another effective model for entity retrieval uses the fielded sequential dependence model (FSDM [27]), which estimates the probability of relevance using information from single terms and bigrams, normalized per field.

2.3 Using entity linking for entity retrieval

Linking entities mentioned in the query to the knowledge graph [3,9] enables the use of relationships encoded in the knowledge graph, helping improve the estimation of relevance of candidate entities. Previous work has shown empirically that entity linking can increase effectiveness of entity retrieval. In [10], for example, entity retrieval has been combined with entity linking to improve retrieval effectiveness over state-of-the-art methods like FSDM.

Our research uses the Tagme entity linker [8] because it is especially suited to annotate short and poorly composed text like the queries we need to link

to. TAGME adds Wikipedia hyperlinks to parts of the text, together with a confidence score.

2.4 Using embeddings for entity retrieval

Very recent work has applied Trans-E graph embeddings to the problem of entity retrieval, and shown consistent but small improvements [15]. However, Trans-E graph embeddings are not a good choice if the graph has 1-to-many, transitive or symmetric relations, which is the case in knowledge graphs [1]. In our research, we also look into improving entity retrieval using graph embeddings, but use the Wikipedia2Vec representation to address these shortcomings.

3 Embedding Based Entity Retrieval

3.1 Graph Embeddings

We base the training of our entity embeddings on Wikipedia2Vec [26,25]. Taking a knowledge graph as the input, Wikipedia2Vec extends the skip-gram variant of Word2Vec [19,18] and learns word and entity embeddings jointly. The objective function of this model is composed of three components. The first component infers optimal embeddings for words W in the corpus. Given a sequence of words $w_1 w_2 ... w_T$ and a context window of size c, the word-based objective function is:

$$\mathcal{L}_w = \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log \frac{\exp(\mathbf{V}_{w_t}^T \mathbf{U}_{w_{t+j}})}{\sum_{w \in W} \exp(\mathbf{V}_{w_t}^T \mathbf{U}_w)}, \tag{1}$$

where matrices $\mathbf{U}$ and $\mathbf{V}$ represent the input and output vector representations, deriving the final embeddings from matrix $\mathbf{V}$.

The two other components of the objective function take the knowledge graph into account. One addition considers a link-based measure estimated from the knowledge graph (i.e., Wikipedia). This measure captures the relatedness between entities in the knowledge base, based on the similarity between their incoming links:

$$\mathcal{L}_e = \sum_{e_i \in \mathcal{E}} \sum_{e_o \in C_{e_i}, e_i \neq e_i} \log \frac{\exp(\mathbf{V}_{e_i}^T \mathbf{U}_{e_o})}{\sum_{e \in \mathcal{E}} \exp(\mathbf{V}_{e_i}^T \mathbf{U}_e)}. \tag{2}$$

Here, C_e denotes entities linked to an entity e, and $\mathcal{E}$ represents all entities in the knowledge graph.

The last addition to the objective function places similar entities and words near each other by considering the context of the anchor text. The intuition is the same as in classic Word2Vec, but here, words in the vicinity of the anchor text have to predict the entity mention. Considering a knowledge graph with anchors A and an entity e the goal is to predict context words of the entity:

$$\mathcal{L}_a = \sum_{e_i \in A} \sum_{w_o \in a(e_i)} \log \frac{\exp(\mathbf{V}_{e_i}^T \mathbf{U}_{w_o})}{\sum_{w \in W} \exp(\mathbf{V}_{e_i}^T \mathbf{U}_w)}, \tag{3}$$

where $a(e)$ gives the previous and next c words of the referent entity e.

These three components (word context, link structure, and anchor context) are then combined linearly into the following objective function:

$$\mathcal{L} = \mathcal{L}_w + \mathcal{L}_e + \mathcal{L}_a. \tag{4}$$

3.2 Re-ranking Entities

Training the Wikipedia2Vec model on a Wikipedia knowledge graph results in a single graph embedding vector for every Wikipedia entity. The next question to answer is how to use these graph embeddings in the setting of entity retrieval.

We propose a two-stage ranking model, where we first produce a ranking of candidate entities using state-of-the-art entity retrieval models (see Section 2.2), and then use the graph embeddings to reorder these entities based on their similarity to the query entities, as measured in the derived graph embedding space.

Following the related work discussed in Section 2.3, we use the TAGME entity linker to identify the entities mentioned in the query. Given input query Q, we obtain a set of linked entities $E(Q)$ and a confidence score $s(e)$ for each entity, which represents the strength of the relationship between the query and the linked entity. We then compute an embedding-based score for every query Q and entity E:

$$F(E, Q) = \sum_{e \in E(Q)} s(e) \cdot cos(\overrightarrow{E}, \overrightarrow{e}), \tag{5}$$

where $\overrightarrow{E}, \overrightarrow{e}$ denote the embeddings vectors for entities E and e.

The rationale for this approach is the hypothesis that relevant entities for a given query are situated close (in graph embedding space) to the query entities identified by the entity linker.

Consider for example the query *"Who is the daughter of Bill Clinton married to."* TAGME links the query to entities BILL CLINTON with a confidence of 0.66, DAUGHTER with a confidence of 0.13, and SAME-SEX MARRIAGE with a confidence score of 0.21. Highly ranked entities then have a large similarity to these entities, where similarity to BILL CLINTON adds more to the score than similarity to DAUGHTER or SAME-SEX MARRIAGE (as the confidence score of BILL CLINTON is higher than the other two). The relevant entities for this query (according to the DBpedia-Entity V2 test collection [12]) are CHELSEA CLINTON, who is Bill Clinton's daughter, and CLINTON FAMILY. We can reasonably expect these entities to have similarity to the linked entities, confirming our intuition.

To produce our final score, we interpolate the embedding-based score computed using Eq. (5) with the score of the state-of-the-art entity retrieval model used to produce the candidate entities in stage one:

$$score_{total}(E, Q) = (1 - \lambda) \cdot score_{other}(E, Q) + \lambda \cdot F(E, Q) \quad \lambda \in [0, 1]. \tag{6}$$

4 Experimental Setup

4.1 Test collection

In our experiments, we used the DBpedia-Entity V2 test collection [12]. The collection consists of 467 queries and relevance assessments for 49280 query-entity pairs, where the entities are drawn from the DBpedia 2015-10 dump. The relevance assessments are graded values of 2, 1, and 0 for highly relevant, relevant, and not relevant entities, respectively. The queries are categorized into 4 different groups: **SemSearch ES** consisting of short and ambiguous keyword queries (e.g., *"Nokia E73"*), **INEX-LD** containing IR-Style keyword queries (e.g., *"guitar chord minor"*), **ListSearch** consisting of queries seeking for a list of entities (e.g., *"States that border Oklahoma"*), and **QALD-2** containing entity-bearing natural language queries (e.g., *"Which country does the creator of Miffy come from"*). Following the baseline runs curated with the DBpedia-Entity V2 collection, we used the stopped version of queries, where stop patterns like "which" and "who" are removed from the queries.

4.2 Embedding Training

Wikipedia2Vec provides pre-trained embeddings. These embeddings, however, are not available for all entities in Wikipedia; e.g., 25% of the assessed entities in DBpedia-Entity V2 collection have no pre-trained embedding. The reasons for these missing embeddings are two-fold: (i) "rare" entities were excluded from the training data, and, (ii) entity identifiers evolve over time, resulting in entity mismatches with those in the DBpedia-Entity collection.

For training new graph embeddings, we used Wikipedia 2019-07 dump. This was the newest version at the time of training. We address the entity mismatch problem by identifying the entities that have been renamed in the new Wikipedia dump. Some of these entities were obtained using the redirect API of Wikipedia.[2] Others were found by matching the Wikipedia page IDs of the two Wikipedia dumps. The page IDs of Wikipedia 2019-07 were available on the Wikipedia website. For the dump where DBpedia-Entity is based on, however, these IDs are not available anymore; we obtained them from the Nordlys package [11].

To avoid excluding rare entities and generate embeddings for a wide range of entities, we changed several Wikipedia2Vec settings. The two settings that resulted in the highest coverage of entities are: (i) minimum number of times an entity appears as a link in Wikipedia, (ii) whether to include or exclude disambiguation pages. Table 1 shows the effect of these settings on the number of missing entities; specifically the number of entities that are assessed in the DBpedia-Entity collection, but have missing embeddings. We categorize these missing entities into two groups:

- *No-page*: Entities without any pages. These entities neither were found by the Wikipedia redirect API nor could be matched by their page IDs.

[2] `https://wikipedia.readthedocs.io/en/latest/`

Table 1: Missing entities with different settings

Settings	No-emb	No-page	Total
min-entity-count = 5, disambiguation = False	9640	608	10248
min-entity-count = 1, disambiguation = False	1220	398	1618
min-entity-count = 1, disambiguation = True	1220	377	1597
min-entity-count = 0, disambiguation = False	**724**	380	1104
min-entity-count = 0, disambiguation = True	**724**	**333**	**1057**

– *No-emb*: Entities that could be found by their identifiers, but were not included in the Wikipedia2Vec embeddings.

The first line in Table 1 corresponds to the default setting of Wikipedia2Vec, which covers only 75% of assessed entities in the DBpedia-Entity collection. When considering all entities in the knowledge graph, this setting discards an even larger number of entities, which is not an ideal setup for entity ranking. By choosing the right settings (the last line of Table 1), we increased the coverage of entities to 97.6%.

We trained two versions of embeddings: with and without link graph; i.e., using Eq. (4) with and without the $\mathcal{L}_e$ component.

4.3 Parameter Setting

Our entity re-ranking approach involves free parameter λ that needs to be estimated (see Eq. (6)). To set this parameter, we employed the Coordinate Ascent algorithm [17] with random restart of 3, optimized for NDCG@100. All experiments were performed using 5-fold cross-validation, where the folds were obtained from the collection (DBpedia-Entity V2). This makes our results comparable to the DBpedia-Entity V2 baseline runs, as the same folds are used for all the methods. Entity re-ranking was performed on top 1000 entities ranked by two state-of-the-art term-based entity retrieval models: FSDM and BM25F-CA [12]. For all experiments, we used the embedding vectors of 100 dimensions, which were trained using the settings described in Section 4.2.

5 Results and Analysis

5.1 Overall Performance

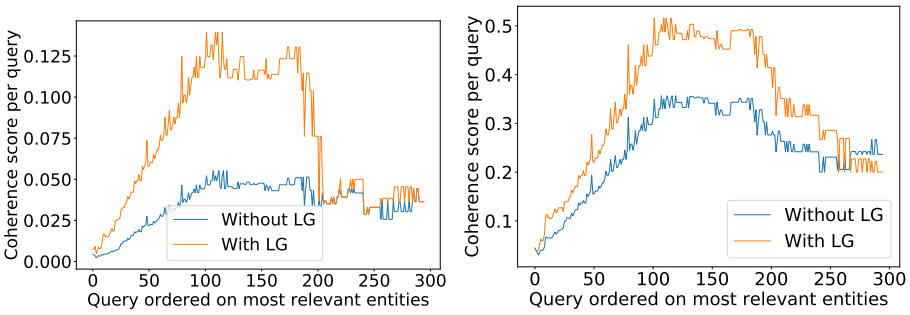
To answer our first research question, whether embeddings improve the score of entity retrieval, we compare our entity re-ranking approach with a number of baseline entity retrieval models. Table 2 shows the results for different models with respect to NDCG@10 and NDCG@100, the default evaluation measures for DBpedia-entity V2. In this table, the embedding-based similarity component (Eq. (5)) is denoted by *ESim*, where *c* and *cg* subscripts refer to the two versions of our entity embeddings: without and with link graph.

Table 2: Results of embedding-based entity re-ranking approach on different query subsets of DBpedia-Entity V2 collection. Significance of results is explained in running text.

Model	**SemSearch**		**INEX-LD**		**ListSearch**		**QALD-2**		**Total**	
NDCG	@10	@100	@10	@100	@10	@100	@10	@100	@10	@100
Reranking the FSDM top 1000 entities										
ESim_c	0.365	0.412	0.194	0.252	0.210	0.288	0.192	0.255	0.239	0.300
ESim_{cg}	0.397	0.462	0.216	0.282	0.211	0.311	0.213	0.286	0.258	0.334
FSDM	0.652	0.722	0.421	0.504	0.420	0.495	0.340	0.436	0.452	0.534
+ELR	0.656	0.726	0.435	0.513	0.422	0.496	0.347	0.446	0.459	0.541
$+\text{ESim}_c$	0.659	0.725	0.433	0.513	0.432	0.509	0.353	0.447	0.463	0.543
$+\text{ESim}_{cg}$	**0.672**	0.733	0.440	0.528	0.424	0.507	0.349	0.451	0.465	0.549
Reranking the BM25F-CA top 1000 entities										
ESim_c	0.381	0.424	0.194	0.253	0.211	0.283	0.192	0.252	0.243	0.301
ESim_{cg}	0.417	0.478	0.217	0.286	0.211	0.302	0.212	0.282	0.262	0.335
BM25F-CA	0.628	0.720	0.439	0.530	0.425	0.511	0.369	0.461	0.461	0.551
$+\text{ESim}_c$	0.658	0.730	0.462	0.545	0.448	0.529	0.380	0.469	0.481	0.563
$+\text{ESim}_{cg}$	0.660	**0.736**	**0.466**	**0.552**	**0.452**	**0.535**	**0.390**	**0.483**	**0.487**	**0.572**

The results of our method are presented for components ESim_c and ESim_{cg} by themselves (i.e., $\lambda = 1$ in Eq. (6)), and also in combination with FSDM and BM25F-CA. The mean and standard deviation of λ found by the Coordinate Ascent algorithm over all folds are: 0.34 ± 0.02 for FSDM+ESim_c, 0.61 ± 0.01 for FSDM+ESim_{cg}, 0.81 ± 0.03 for BM25F-CA+ESim_c, and 0.88 ± 0.00 for BM25F-CA+ESim_{cg}. The results show that the embedding-based scores alone do not perform very well, however, when combining them with other scores, the performance improves by a large margin. We determine the statistical significance of the difference in effectiveness for both the NDCG@10 and the NDCG@100 values, using the two-tailed paired t-test with $\alpha < 0.05$. The results show that both versions of FSDM+ESim and BM25-CA+ESim models yield significant improvements over FSDM and BM25-CA models (with respect to all metrics), respectively. Also, FSDM+ESim_{cg} improves significantly over FSDM+ELR with respect to NDCG@100, showing that our embedding based method captures entity similarities better than the strong entity ID matching approach used in the ELR method.

When considering the query subsets, we observe that FSDM+ESim_{cg} significantly outperforms FSDM for SemSearch and QALD queries with respect to NDCG@10, and for INEX-LD queries with respect to NDCG@100. Improvements over BM25F-CA were more substantial: BM25F-CA+ESim_{cg} brings significant improvements for all categories (with respect to all metrics) except for SemSearch queries for NDCG@100.



(a) threshold $\tau = 0.9$ (b) threshold $\tau = 0.8$

Fig. 1: Coherence score of all relevant entities per query, computed for the versions of entity embeddings (without and with link graph) . The queries are ordered by the number of their relevant entities in x-axis.

5.2 Entity Embeddings Analysis

The results of Table 2 suggest that graph-based entity embeddings yield better performance compared to context only entity embeddings. To analyze why graph-based entity embeddings are beneficial for entity retrieval models, we conduct a set of experiments and investigate properties of embeddings with and without the graph structure.

According to the cluster hypothesis [14], documents relevant to the same query should cluster together. We consider the embeddings as data-points to be clustered and compare the resulting clusters in several ways. First, we compute the Davies Bouldin index [5] and the Silhouette index [24], which are: 3.16 and 0.08 for the embeddings with link graph, and 3.98 and -0.05 for the embeddings without link graph, respectively. Both measures indicate that better clusters arise for the embeddings that capture graph structure.

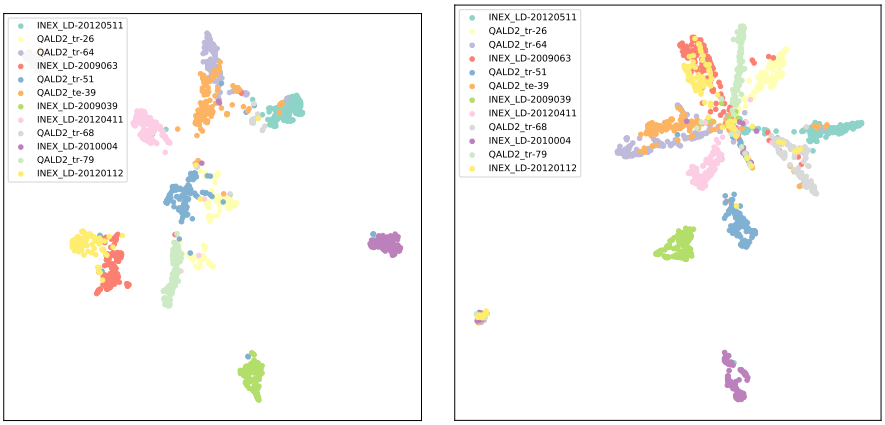
To get an indication of how coherent the clusters are, we compute for each query the coherence score defined in [13]. This score measures the similarity between item pairs of a cluster and returns the percentage of items with similarity score higher than a threshold, thereby assigning high scores to the clusters that are coherent. Formally, given a document set D, the coherence score is computed as:

$$Co(D) = \frac{\sum_{i \neq j \in 1,\ldots,M} \delta(d_i, d_j)}{\frac{1}{2}M(M-1)}, \tag{7}$$

where M is total number of documents and the δ function for each document pair d_i and d_j is defined as:

$$\delta(d_i, d_j) = \begin{cases} 1, & \text{if } sim(d_i, d_j) \geq \tau \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

We compute the coherence score with thresholds $0.8, 0.9$, using *cosine* for similarity function $sim(d_i, d_j)$, where d_i and d_j correspond to entities. Figure 1



(a) Embeddings with link graph (b) Embeddings without link graph

Fig. 2: UMAP visualization of entity embeddings for a subset of queries. Color-codes correspond to the relevant entities per query. Queries per code are listed in Table 5 of the Appendix. Default settings of UMAP in python were used.

shows the results of coherence score for all queries in our collection. Each point represents the coherence score of all relevant entities (according to the qrels) for a query. We considered only queries with more than 10 relevant entities, for clusters large enough to compute a meaningful score. Queries are sorted on the x-axis by the number of relevant entities. The plots clearly show that the coherence score for graph-based entity embeddings is higher than for context only ones. Based on these performance improvements we conclude that adding the graph structure results in embeddings that are more suitable for entity-oriented tasks.

Figures 2 helps to visually understand how clusters of entities differ for the two methods (a subset of all entities is shown for clarity). The data points correspond to the entities with a relevance grade higher than 0, for 12 queries with 100–200 relevant entities in the ground truth data. We use Uniform Manifold Approximation and Projection (UMAP) [16] to reduce the embeddings dimensions from 100 to two and plot the projected entities for each query. In Figure 2b most of the clusters are overlapping in a star-like shape, while in Figure 2a the clusters are more separated and the ones with similar search intents are close to each other; e.g., queries `QALD2_te-39` and `QALD2_tr-64` (which are both about companies), or `INEX_LD-20120112` and `INEX_LD-2009063` (which are both about war) are situated next to each other. To observe how false positive entities are placed in the embedding space, we added the 10 highest ranked false positives to the data and created new UMAP plots. In the obtained plots, false positive entities that are semantically similar to the true positive entities are close to each other. For example, two false positive entities for the query *"South Korean girl groups"* are: SHINee (a South Korean boy band) and Hyuna (a South Korean female singer). Both of these entities are semantically similar to the relevant en-

Table 3: Top queries with the highest gains and losses in NDCG at cut-offs 10 and 100, BM25F + ESim$_{cg}$ vs. BM25F.

Query	Gain in NDCG	
	@10	@100
st paul saints	0.716	0.482
continents in the world	0.319	0.362
What did Bruce Carver die from?	0.307	0.307
spring shoes canada	-0.286	-0.286
vietnam war movie	-0.470	-0.240
mr rourke fantasy island	-0.300	-0.307

Table 4: Top queries with the highest gains and losses in NDCG at cut-offs 10 and 100, BM25F + ESim$_{cg}$ vs. BM25F + ESim$_{c}$.

Query	Gain in NDCG	
	@10	@100
What did Bruce Carver die from?	0.307	0.307
Which other weapons did the designer of the Uzi develop?	0.236	0.248
Which instruments did John Lennon play?	0.154	0.200
Companies that John Hennessey serves on the board of	-0.173	-0.173
Which European countries have a constitutional monarchy?	-0.101	-0.197
vietnam war movie	-0.276	-0.222

tities of the query and are also placed in the vicinity of them, although they do not address the information needs of the query. This is consistent with the plots of Figure 2 and in line with our conclusion on the effect of graph embeddings for entity-oriented search.

5.3 Query Analysis

Next, we investigate our second research question and analyse queries that are helped and hurt the most by our embedding-based method. Table 3 shows six queries that are affected the most by BM25F-CA+ESim$_{cg}$ compared to BM25F-CA (on NDCG@100). Each of the three queries with highest gains are linked to at least one relevant entity (according to the assessments). The losses can be attributed to various sources of errors. For the query *"spring shoe canada"*, the only relevant entity belongs to the 2.4% of entities that have no embedding (cf. §4.2). Query *"vietnam war movie"* is linked to entities Vietnam War and War film, with confidence scores of 0.7 and 0.2, respectively. This emphasizes Vietnam war facts instead of its movies, and could be resolved by improving the accuracy of the entity linker and/or employing a re-ranking approach that is more robust to linking errors. The query *"mr rourke fantasy island"* is linked to a wrong entity due to a spelling mistake. To conclude, errors in entity linking form one of the main reasons of performance loss in our approach.

To further understand the difference between the two versions of the embeddings at the query-level, we selected the queries with the highest and lowest gain in NDCG@100 (i.e., comparing BM25F+ESim$_{cg}$ and BM25F+ESim$_{c}$). For the query *"Which instruments did John Lennon play?"*, the two linked entities (with

the highest confidence score) are John Lennon and Musical Instruments. Their closest entity in graph embedding space is John Lennon's musical instruments, relevant to the query. This entity, however, is not among the most similar entities when we consider the context-only case.

For the other queries in Table 4, the effect is similar but less large than in the BM25F and BM25F + ESim$_{cg}$ case, probably due to the lower value of λ.

6 Conclusion

We investigated the use of entity embeddings for entity retrieval. We trained entity embeddings with Wikipedia2Vec, combined these with state-of-the-art entity ranking models, and find empirically that using graph embeddings leads to increased effectiveness of query results on DBpedia-Entity V2.

The empirical findings can be interpreted as evidence for the cluster hypothesis. Including a representation of the graph structure in the entity embeddings leads to better clusters and higher effectiveness of retrieval results. We further see that queries which get linked to relevant entities or pages neighboring to relevant entities get helped the most, while queries with wrongly linked entities are helped the least.

We conclude that enriching entity retrieval methods with entity embeddings leads to improved effectivenss, but acknowledge the following limitations of this study. Not all query categories lead to improvements on NDCG. While the state-of-the-art in entity-linking has made significant progress in recent years, we applied Tagme to identify the entities in queries. As we observed that lower performance of queries can often be attributed to erroneously linked entities, we expect better results by replacing this component for a state-of-the-art approach. Finally, we have only experimented using the embeddings constructed by Wikipedia2Vec, and plan to continue our experiments using alternative entity embedding methods like TransE.

A Queries

Table 5: Queries mentioned by their query ID.

Query ID	Query text
INEX_LD-20120511	female rock singers
QALD2_tr-26	Which bridges are of the same type as the Manhattan Bridge?
QALD2_tr-64	Which software has been developed by organizations founded in California?
INEX_LD-2009063	D-Day normandy invasion
QALD2_tr-51	Give me all school types.
QALD2_te-39	Give me all companies in Munich.
INEX_LD-2009039	roman architecture
INEX_LD-20120411	bicycle sport races
QALD2_tr-68	Which actors were born in Germany?
INEX_LD-2010004	Indian food
QALD2_tr-79	Which airports are located in California, USA?
INEX_LD-20120112	vietnam war facts

References

1. Heiko Paulheim: Machine learning & embeddings for large knowledge graphs. URL: https://www.slideshare.net/heikopaulheim/machine-learning-embeddings-for-large-knowledge-graphs (7 2019)
2. Balog, K.: Entity-oriented search. Springer (2018)
3. Blanco, R., Ottaviano, G., Meij, E.: Fast and space-efficient entity linking in queries. Proceedings of the Eighth ACM International Conference on Web Search and Data Mining pp. 179–188 (2015)
4. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Proceedings of the 26th International Conference on Neural Information Processing Systems. pp. 2787–2795. ACM (2013)
5. Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE transactions on pattern analysis and machine intelligence **1**(2), 224–227 (1979)
6. Dehghani, M., Zamani, H., Severyn, A., Kamps, J., Croft, W.B.: Neural ranking models with weak supervision. Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval pp. 65–74 (2017)
7. Diaz, F., Mitra, B., Craswell, N.: Query expansion with locally-trained word embeddings. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics pp. 367–377 (2016)
8. Ferragina, P., Scaiella, U.: Tagme: on-the-fly annotation of short text fragments (by Wikipedia entities). In: Proceedings of the 19th ACM international IW3C2 on Information and Knowledge Management. pp. 1625–1628. ACM (2010)
9. Hasibi, F., Balog, K., Bratsberg, S.E.: Entity linking in queries: Tasks and evaluation. In: Proceedings of the 2015 International Conference on The Theory of Information Retrieval. p. 171–180 (2015)
10. Hasibi, F., Balog, K., Bratsberg, S.E.: Exploiting entity linking in queries for entity retrieval. In: Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval. pp. 209–218. ACM (2016)
11. Hasibi, F., Balog, K., Garigliotti, D., Zhang, S.: Nordlys: A toolkit for entity-oriented and semantic search. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1289–1292. ACM (2017)
12. Hasibi, F., Nikolaev, F., Xiong, C., Balog, K., Bratsberg, S.E., Kotov, A., Callan, J.: DBpedia-Entity V2: A test collection for entity search. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1265–1268. ACM (2017)
13. He, J., et al.: Exploring topic structure: Coherence, diversity and relatedness. SIKS (2011)
14. Jardine, N., van Rijsbergen, C.J.: The use of hierarchic clustering in information retrieval. Information storage and retrieval **7**(5), 217–240 (1971)
15. Liu, Z., Xiong, C., Sun, M., Liu, Z.: Explore entity embedding effectiveness in entity retrieval. arXiv preprint arXiv:1908.10554 (2019)
16. McInnes, L., Healy, J., Saul, N., Grossberger, L.: Umap: Uniform manifold approximation and projection. The Journal of Open Source Software **3**(29), 861 (2018)
17. Metzler, D., Bruce Croft, W.: Linear feature-based models for information retrieval. Information. Retrieval **10**(3), 257–274 (jun 2007)

18. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: 1st International Conference on Learning Representations, ICLR. pp. 1–12 (2013)
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems (NIPS). pp. 3111–3119 (2013)
20. Noy, N., Gao, Y., Jain, A., Narayanan, A., Patterson, A., Taylor, J.: Industry-scale knowledge graphs: Lessons and challenges. Communications of the ACM **62**(8), 36–43 (2019)
21. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. pp. 1532–1543. ACL (2014)
22. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international Conference on Knowledge discovery and data mining. pp. 701–710. ACM (2014)
23. Robertson, S., Zaragoza, H., et al.: The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends® in Information Retrieval **3**(4), 333–389 (2009)
24. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics **20**, 53–65 (1987)
25. Yamada, I., Asai, A., Shindo, H., Takeda, H., Takefuji, Y.: Wikipedia2vec: An optimized tool for learning embeddings of words and entities from wikipedia. arXiv preprint 1812.06280 (2018)
26. Yamada, I., Shindo, H., Takeda, H., Takefuji, Y.: Joint learning of the embedding of words and entities for named entity disambiguation. The SIGNLL Conference on Computational Natural Language Learning (2016)
27. Zhiltsov, N., Kotov, A., Nikolaev, F.: Fielded sequential dependence model for adhoc entity retrieval in the web of data. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 253–262. ACM (2015)